CLAIMS

What is claimed is:

1. A method of generating code to be deployed in an application server, comprising the steps of:

receiving an archive file to be deployed;

introspecting an input class included in the archive file to generate information relating to the input class;

generating a markup language description of the input class based on the generated information relating to the input class;

creating an event handler for method nodes found in the markup language description;

registering the event handler;

parsing the markup language description and invoking the registered event handler; and

generating output code using the invoked event handler.

2. The method of claim 1, wherein the class file is an Enterprise Java Bean class file.

3. The method of claim 2, wherein the step of introspecting an input class included in the archive file comprises the steps of:

extracting information identifying methods included in the input class; and

for each method, extracting information relating to parameters of the method.

4. The method of claim 3, wherein the step of generating a markup language description of the input class comprises the step of:

generating an Extensible Markup Language description of the input class based on the generated information relating to the input class.

5. The method of claim 4, wherein the step of creating an event handler comprises the step of:

creating a Simple Application Programming Interface for Extensible Markup Language event handler for a method node found in the markup language description

6. The method of claim 5, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the markup language description using a Simple Application Programming Interface for Extensible Markup Language parser and invoking the registered Simple Application Programming Interface for Extensible Markup Language event handler.

7. The method of claim 1, wherein the step of creating an event handler comprises the step of:

creating a plurality of event handlers for a method node found in the markup language description.

8. The method of claim 7, wherein the step of registering the event handler comprises the step of:

registering each of the plurality of event handlers.

9. The method of claim 8, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the markup language description and invoking each of the plurality of registered event handlers.

10. The method of claim 9, wherein the step of generating output code comprises the step of:

generating output code using each of the plurality of invoked event handler in parallel.

11. The method of claim 10, wherein the archive file is an Enterprise Java Bean archive file.

12. The method of claim 11, wherein the step of introspecting an input class included in the archive file comprises the steps of:

extracting information identifying methods included in the input class; and

for each method, extracting information relating to parameters of the method.

13. The method of claim 12, wherein the step of generating a markup language description of the input class comprises the step of:

generating an Extensible Markup Language description of the input class based on the generated information relating to the input class.

14. The method of claim 13, wherein the step of creating a plurality of event handlers comprises the step of:

creating a plurality of Simple Application Programming Interface for Extensible Markup Language event handlers for a method node found in the markup language description

15. The method of claim 14, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the markup language description using a Simple Application Programming Interface for Extensible Markup Language parser and invoking the plurality of registered Simple Application Programming Interface for Extensible Markup Language event handlers.

16. A system for generating code to be deployed in an application server comprising:

a processor operable to execute computer program instructions;

a memory operable to store computer program instructions executable by the processor; and

computer program instructions stored in the memory and executable to perform the steps of:

receiving an archive file to be deployed;

introspecting an input class included in the archive file to generate information relating to the input class;

generating a markup language description of the input class based on the generated information relating to the input class;

creating an event handler for a method node found in the markup language description;

registering the event handler;

parsing the markup language description and invoking the registered event handler; and

generating output code using the invoked event handler.

17. The system of claim 16, wherein the archive file is an Enterprise Java Bean archive file.

18. The system of claim 17, wherein the step of introspecting an input class included in the archive file comprises the steps of:

extracting information identifying systems included in the input class; and

for each system, extracting information relating to parameters of the system.

19. The system of claim 18, wherein the step of generating a markup language description of the input class comprises the step of:

generating an Extensible Markup Language description of the input class based on the generated information relating to the input class.

20. The system of claim 19, wherein the step of creating an event handler comprises the step of:

creating a Simple Application Programming Interface for Extensible Markup Language event handler for a system node found in the markup language description

21. The system of claim 20, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the markup language description using a Simple Application Programming Interface for Extensible Markup Language parser and invoking the registered Simple Application Programming Interface for Extensible Markup Language event handler.

22. The system of claim 21, wherein the step of creating an event handler comprises the step of:

creating a plurality of event handlers for a system node found in the markup language description.

23. The system of claim 22, wherein the step of registering the event handler comprises the step of:

registering each of the plurality of event handlers.

24. The system of claim 23, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the markup language description and invoking each of the plurality of registered event handlers.

25. The system of claim 24, wherein the step of generating output code comprises the step of:

generating output code using each of the plurality of invoked event handler in parallel.

26. The system of claim 25, wherein the archive file is an Enterprise Java Bean archive file.

27. The system of claim 26, wherein the step of introspecting an input class included in the archive file comprises the steps of:

extracting information identifying systems included in the input class; and

for each system, extracting information relating to parameters of the system.

28. The system of claim 27, wherein the step of generating a markup language description of the input class comprises the step of:

generating an Extensible Markup Language description of the input class based on the generated information relating to the input class.

29. The system of claim 28, wherein the step of creating a plurality of event handlers comprises the step of:

creating a plurality of Simple Application Programming Interface for Extensible Markup Language event handlers for a system node found in the markup language description

30. The system of claim 29, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the markup language description using a Simple Application Programming Interface for Extensible Markup Language parser and invoking the plurality of registered Simple Application Programming Interface for Extensible Markup Language event handlers.

31. A computer program product for generating code to be deployed in an application server comprising:

a computer readable medium;

computer program instructions, recorded on the computer readable medium, executable by a processor, for performing the steps of

receiving an archive file to be deployed;

introspecting an input class included in the archive file to generate information relating to the input class;

generating a markup language description of the input class based on the generated information relating to the input class;

creating an event handler for a method node found in the markup language description;

registering the event handler;

parsing the markup language description and invoking the registered event handler; and

generating output code using the invoked event handler.

32. The computer program product of claim 31, wherein the archive file is an Enterprise Java Bean archive file.

33. The computer program product of claim 32, wherein the step of introspecting an input class included in the archive file comprises the steps of:

extracting information identifying computer program products included in the input class; and

for each computer program product, extracting information relating to parameters of the computer program product.

34. The computer program product of claim 33, wherein the step of generating a markup language description of the input class comprises the step of:

generating an Extensible Markup Language description of the input class based on the generated information relating to the input class.

35. The computer program product of claim 34, wherein the step of creating an event handler comprises the step of:

creating a Simple Application Programming Interface for Extensible Markup Language event handler for a computer program product node found in the markup language description

36. The computer program product of claim 35, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the markup language description using a Simple Application Programming Interface for Extensible Markup Language parser and invoking the registered Simple Application Programming Interface for Extensible Markup Language event handler.

37. The computer program product of claim 31, wherein the step of creating an event handler comprises the step of:

creating a plurality of event handlers for a computer program product node found in the markup language description.

38. The computer program product of claim 37, wherein the step of registering the event handler comprises the step of:

registering each of the plurality of event handlers.

39. The computer program product of claim 38, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the markup language description and invoking each of the plurality of registered event handlers.

40. The computer program product of claim 39, wherein the step of generating output code comprises the step of:

generating output code using each of the plurality of invoked event handler in parallel.

41. The computer program product of claim 40, wherein the archive file is an Enterprise Java Bean archive file.

42. The computer program product of claim 41, wherein the step of introspecting an input class included in the archive file comprises the steps of:

extracting information identifying computer program products included in the input class; and

for each computer program product, extracting information relating to parameters of the computer program product.

43. The computer program product of claim 42, wherein the step of generating a markup language description of the input class comprises the step of:

generating an Extensible Markup Language description of the input class based on the generated information relating to the input class.

44. The computer program product of claim 43, wherein the step of creating a plurality of event handlers comprises the step of:

creating a plurality of Simple Application Programming Interface for Extensible Markup Language event handlers for a computer program product node found in the markup language description

45. The computer program product of claim 44, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the markup language description using a Simple Application Programming Interface for Extensible Markup Language parser and invoking the plurality of registered Simple Application Programming Interface for Extensible Markup Language event handlers.